IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

Detecting Phishing Attacks Using Natural Language Processing and Deep Learning models

[1]Fenny Zalavadia, [2]Shubhangi Pandey, [3]Priyanka Pachpande, [4]Akshata Nevrekar, [5]Dr. Sharvari Govilkar
Student,Department of Computer Engineering, Mumbai University, PCE, New Panvel, India

Abstract: Phishing attacks are one of the most prevalent and least defended security threats today. An approach is proposed which uses Natural Language Processing (NLP) techniques to analyze text and detect inappropriate statements that are indicative of phishing attacks. NLP offers a wholesome solution for this problem as it is capable of analyzing the textual content to perform intelligent recognition and performing semantic analysis of text to detect malicious intent. The approach also makes use of Deep Learning frameworks with Neural Network. The Phishector (Phishing Detector) takes input as an email and gives the output as the accuracy with which it is predicting whether an email is phished or legitimate. We have worked upon 2 datasets: SpamAssassin and Ham-Spam datasets. Also build our model using various Machine Learning techniques such as Bagged decision tree, Random Forest, Extra Trees, Adaboost, Stochastic Gradient Descent, Naive Bayes, SVM, and Voting Ensemble. As per our observance, Random Forest and Extra Trees outperform for SpamAssassin and HSD dataset respectively. Neural Network also outperforms when tested with a variety of layers, epoch, and batch size using Adam optimizer.

1. Introduction

Phishing occurs when cybercriminals send malicious emails specifically designed to deceive individuals into tripping for a scam. The main aim is to impulse the users to divulge their financial data, credential or sensitive information. “Phishing” the term came about in the mid-1990s, when fraudsters began to use fraudulent emails to obtain information from unsuspecting users. Cybercriminals use phishing technique as it is simple, low cost and effective. Obtaining email addresses are easier and free to send. With some effort and small price, attackers will quickly gain access to treasure information. We can detect these emails and detect them as phished and reduce these attacks. To achieve this we can make use of various machine learning and deep learning techniques.

In 2003, Paypal users were hit by the virus named “Mimail”, a forged popup window from Paypal requesting ID and Password will be opened when one clicked the link. Which then was immediately sent to the hackers.

In 2004, John Kerry received legitimate looking email, convincing him to donate via an enclosed link provided in the email. Later, it was revealed that due to no affiliation towards Kerry campaign someone from India and Texas had set up this scam.

Now, the strategies to detect Phishing emails are as diverse as fish within the sea, fraudsters still come up with new strategies to achieve trust, bring disturbance and avoid detection. One among several troubling trends is that the usage of data obtained through social sites to establish the communications as personal as possible, generally cited as “spear-phishing” or “social engineering fraud.”

2. Literature Survey

2.1. Random Forest:

Andronicus A. Akinyelu, Aderemi O. Adewumi proposed a classifier that has fewer numbers of features and improved prediction accuracy. From a dataset comprising 2000 phishing and legitimate emails, a collection of essential phishing email features were extracted and applied to the machine learning algorithm for classification having an accuracy of 99.7% [2]. Srishti Rawal, Bhuvan Rawal, Aakhila Shaheen, Shubham Malik discussed a phished email classifier made using self-made dataset with ‘n’ phished mails and ‘m’ ham mails[9] which extracted 9 features from this dataset.

2.2. Support Vector Machine:

Fergus Toolan, Joe Carthy [1] the instances provided by him are very little consisting of only 5 features. They evaluated the results of this system, comprising 8,000 emails out of which approximately half were phishing emails and remaining were legitimate emails. Adwan Yasin, Abdel Munem Abuhasan [6] proposed a model that applied the knowledge discovery procedures with five favoured classification algorithms among which SVM was one and achieved an exceptional enhancement in classification accuracy. Srishti Rawal, Bhuvan Rawal, Aakhila Shaheen, Shubham Malik [9] goal was to use the minimal number of features to develop a model which provides greater accuracy and to study the variation of features. The features were extracted using regular expressions and NLTK. 99.87% of accuracy is achieved using SVM model to classify emails. This accuracy was the maximum.

2.3. Naive Bayes:

Adwan Yasin, Abdel Munem Abuhasan [6] proposes the concept of phishing terms, which estimates the weight of phishing terms in each email. By applying stemming and WordNet ontology the pre-processing phase is strengthened to enhance the model with word synonyms. Elif Yerli, Ibrahim Sogukpinar [7] discussed a technique from which a success rate of 89% has been achieved against phishing attacks coming from email messages. Srishti Rawal, Bhuvan Rawal, Aakhila Shaheen, Shubham Malik discussed a phished email classifier made using self-made dataset with ‘n’ phished mails and ‘m’ ham mails[9] which extracted 9 features from this dataset.These features are given to the classifiers and result was noted.

2.4. LSTM:

Minh Nguyen, Toan Nguyen, Thien Huu Nguyen [10] presented a framework with H-LSTMs memory networks and attention mechanisms to model the emails simultaneously at the word and sentence level. To generate an effective model for anti-phishing and establish the effectiveness of deep learning for problems in cybersecurity is expected.The performance evaluator used are precision, recall and F1-score to calculate the performance of the models for detecting phishing emails and compare it with SVM baseline models in two different settings first, when the email headers are ignored. Second, types of data: without header and with header. Without header accuracy of 98.1% and with header accuracy of 99%.

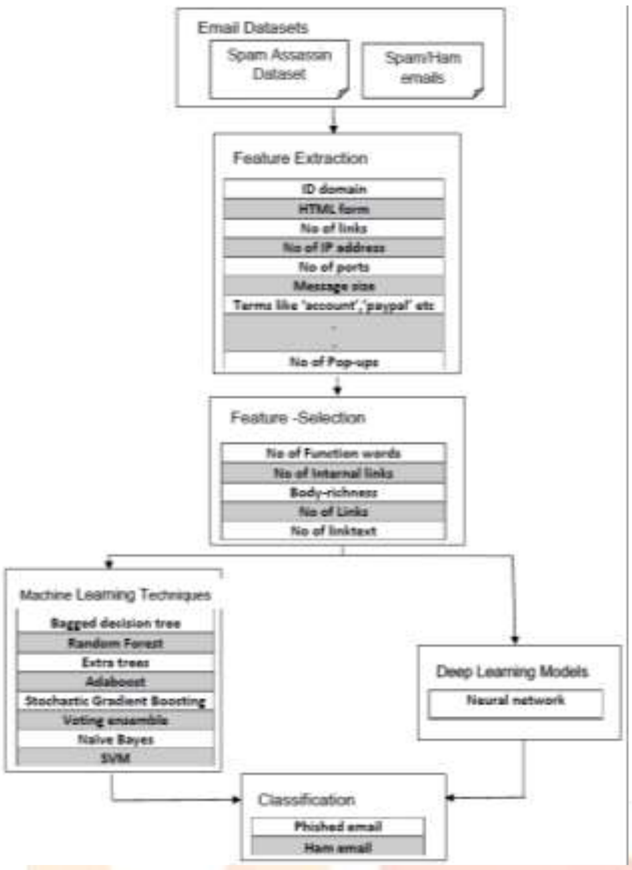
2.5 Summary of Related Work

After going through most of the research papers from 2014 to 2018 on the topic Email Phishing detection we can infer that mostly the dataset that are used are Spamassassin and Phishing Corpus and these are widely open sourced dataset and easily available. The ML techniques mostly used till date are SVM, Random Forest, Naïve Bayes, Logistic Regression, Clustering and the latest research papers have used DL techniques such as Neural networks and LSTM. A few of the papers have encountered a high accuracy but on a small set of data. In 2014 Paper Clustering technique was used, which has acquired good clusters but interpreting those cluster behavior is a bit difficult. Whereas on the other hand we can observe that algorithms such as SVM, Random Forest have outperformed on various datasets and provided accuracy above 99%.

3. Proposed Architecture

In this chapter we would be discussing the system architecture. The previous sections discussed the strengths and weaknesses of existing system. In order to achieve better domain results, researchers combined both techniques to build Hybrid domain systems, which seek to inherit advantages and eliminate disadvantages.

Figure 3.1 Proposed architecture



3.1 Email Dataset

For training our model we have used Spamassassin and HAM dataset. Spamassassin open source anti-spam platform giving system administrators a filter to classify email and block spam. Spam/Ham dataset is an open source dataset available on Kaggle. Spamassassin dataset has 1795 spam emails and 5051 ham emails. Spam/Ham dataset has 481 spam emails and 481 ham emails.

3.2 Feature Extraction

Initially we have extracted 40 features that play a role in categorizing an email into phished or ham. These features return Boolean or integer values which are used in further detection.

Table 3.2: Extracted features

Features	*Meaning*
Body_forms	It returns true if html has <form>.
body_html	It returns true if html is present.
body_noCharacters	It returns the no. of characters in the body.
body_noDistinctWords	It returns no. of distinct words
body_noFunctionWords	It returns a no. of function words.
body_noWords	It returns the no. of words in the body.

Body_richness	It calculates and returns the richness of the text in the body.
Body_suspension	It returns TRUE if the body has the word "suspension".
body_verifyYourAccount	It returns TRUE if the body has the phrase "verify your account".
script_javaScript	It returns TRUE if the script is present in javascript.
script_noOnClickEvents	It returns a no. of onclick events.
script_nonModalJsLoads	It returns TRUE if Javascript comes from outside the modal Domain.
script_popups	It returns TRUE, if email contains pop-up window code.
Script_scripts	It returns TRUE if scripts are present in the email body.
script_statusChange	It returns TRUE if the script overrides the status bar in the email client.
send_diffSenderReplyTo	It returns TRUE if sender and reply-to domain is different.
send_noCharacters	It returns the no. of characters in the sender's address.
send_noWords	It returns the no. of words in the sender's address.
send_nonModalSenderDomain	It returns TRUE, if the sender's and emails modal domain are different.
subj_bank	It returns TRUE if subject has the word "bank"
Subj_debit	It returns TRUE if the subject has the word "debit".
subj_forward	It returns TRUE if email is being forwarded.
subj_no_Characters	It returns the no. of characters in the subject.

subj_no_Words	It returns the no. of words in the subject.
Subj_reply	It returns TRUE if email is being replied to any of the previous mails.
Subj_richness	It calculates and returns the richness of the subject.
subj_verify	It returns TRUE if the body has the word “verify”.
url_atSymbol	It returns TRUE if @ symbol is present in url.
url_ipAddress	It returns TRUE if there is use of Ip address instead of domain name.
url_linkText	It returns TRUE if link text contains click, here, login or update terms.
url_max_NoPeriods	It returns no. of periods in the link with the highest no. of periods.
url_noDomains	It returns the no. of url domains in the email body.
url_noExtLinks	It returns a no. of external links in the email body.
url_noImgLinks	It returns a no. of image links in the email body.
url_noIntLinks	It returns a no. of internal links in the email body.
url_noIpAddresses	It returns a no. of links in email that contain Ip address.
url_noLinks	It returns a no. of links in the email body.
url_noPorts	It returns a no. of links with port information.
url_nonModalHereLinks	It returns TRUE if 'here' links don't map to the modal domain.
Url_ports	It returns TRUE if the URL accesses ports other than 80.

3.3 Feature Selection

Feature selection is the process where it is viable to get those features in data which are of most use and relevance. Feature selection methods help to develop accurate predictive models. They assist in selecting features that provide better accuracy although requiring fewer data. These methods can be helpful in identifying and eliminating unwanted, irrelevant and redundant attributes from data which don't add much to the accuracy of a predictive model or indeed reduce the accuracy of the model. Fewer attributes are required since it reduces the complexity of the model, and a simpler model is easier to understand and describe. Three-fold is used for variable selection: improving the prediction results, providing much faster and more cost-effective predictors, and providing a better understanding of the underlying process that generated the data.

mRMR (minimum Redundancy Maximum Relevance):

The mRMR (minimum Redundancy Maximum Relevance) approach is used to expand the joint dependency of top ranking features on the target test dataset, at the same time the redundancy among the test cases must be lessened. This is done by incrementally selecting the maximally relevant test cases from the test dataset with respect to the features while avoiding the redundant data. For this initially, the mutual information (MI) between the test cases and the feature is calculated (the relevance term). Then the average MI between the test cases and the remaining test cases that are already selected is computed. We have found out that for different datasets different sets of features are important. For spamassassin dataset there are 5 features selected and for HSD there are 8 features selected. After reducing the number of features from 40 to 5 and 8 there is not much difference in accuracy. So we use only selected features for further processing instead of using all 40 features.

For Spamassassin:

1. Body_noFunctionWords
2. url_noIntLinks
3. body_richness
4. url_noLinks
5. url_linkText

For HSD:

1. body_richness
2. subj_richness
3. body_forms
4. body_html
5. body_noFunctionWords
6. body_verifyYourAccount
7. script_javaScript
8. script_noOnClickEvents

3.4 Machine Learning Techniques

3.4.1 Naive Bayes Classifier

It is a classification technique based on Bayes' Theorem with a presumption of independence between predictors. Simply a Naive Bayes classifier presumes that the existence of a particular feature in a class is unrelated to the existence of any other feature. Even if these features depend upon the existence of the other features, all of these properties independently contribute to the probability. Naive Bayes model is useful for very huge datasets, also it is easier to build. Along with lucidity, Naive Bayes is well known to outperform even on highly advanced classification methods. We provide input in terms of selected features to naive Bayes' model where the model predicts the probability of a data point belonging to a phished or legitimate class.

3.4.2 Random Forest

It is an ensemble learning method for regression, classification, etc . It works by constructing a bunch of decision trees at training time and predicting the class whether it is the classification or regression of the individual trees. It corrects for decision trees' habit of overfitting to their training set. Input to random forest classifier is selected features wherein a tree is built using edges as criteria to belong to phished or legitimate class and nodes represent the important features in a specific order.

3.4.3 Support Vector Machine

SVM is supervised learning models used for analysing regression and classification datasets. Having a set of training data-points, each labelled as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new data-points to one category or the other, making it a non-probabilistic binary linear classifier. It takes selected features as input and tries to find the best fit hyperplane which separates phished and legitimate with maximum margin distance.

3.4.4 Bagged Decision Tree

Bootstrap Aggregation (Bagging), is a simple and very powerful ensemble method. It is a technique that combines the predictions from multiple machine learning algorithms (SVM, Naive Bayes, Decision tree, etc.) jointly to make more accurate predictions than any individual model. Decision trees are sensitive to the specific data on which they are trained. If the training data is changed the resulting decision tree can be slightly different and in turn the Predictions can be quite different. Bootstrap Aggregation is a conventional procedure that can be used to reduce the variance for those algorithms that have high variance. Input to Bagged decision tree classifier is selected features wherein a tree is built using edges as criteria to belong to phished or legitimate class and nodes represent the important features in a specific order. Here trees are not pruned hence grow deep.

3.4.5 Extra Trees

The aim of Extra-Tree (extremely randomized trees) is to further randomize the tree building in the context of numerical input features, where the selection of the optimal cut point is responsible for a large part of the variance of the induced tree. In contrast to random forests, extra-trees drop the idea of using bootstrap strategy of learning samples, instead it tries to find an optimal cut-point for each one of the K randomly chosen features at each node, it picks a cut-point at random. Input to extra tree classifier is selected features wherein a tree is built using edges as criteria to belong to phished or legitimate class and nodes represent the important features in a specific order.

3.4.6 Adaboost

To enhance the performance and increase the accuracy of decision trees on binary classification problems adaboost are used. It performs best when used with weak learners. On classification problems Adaboost achieves accuracy above random model. Decision trees with one level are more suited and frequently used algorithm with adaboost because these trees are so short and only contain one decision for classification, they are often called decision stumps. By computing the weighted average of the weak classifiers predictions are made.

3.4.7 Voting Ensemble

Voting ensemble method integrates the predictions of various base classifiers and provides the final prediction. The ensemble model integrates the set of classifiers to create a single composite model providing better accuracy. The aggregating technique that combines the results of multiple classifiers is called "Voting". There are 3 versions of voting namely "unanimous voting", "plurality voting" and "majority voting". In unanimous voting all classifiers accept the final decision, plurality voting majority voting is considered consisting of more than 50% vote for final decision. We have used Majority voting for prediction in our model. We have used SVM, Naive Bayes, Random forest, Decision tree and adaboost for classification and for final decision majority vote to the classifier is taken into consideration for classification.

3.4.8 Stochastic Gradient Boosting

The word "Stochastic" in the SGB (Stochastic Gradient Boosting) algorithm, an arbitrary percentage of training data is used for every iteration instead of using all data for training. This results in improvement to previous outcomes. In order to upgrade the accuracy of a predictive function by applying the function frequently in a series and combining the output of each function. Boosting is preferred. It consists of a Shrinkage factor where if each tree in the series is multiplied by this factor ranging from 0 to 1, it will delay the learning process and accordingly, the length of the series will be longer to compensate for the shrinkage, thus resulting in better prediction value.

3.5 Deep Learning Techniques

3.5.1 Neural Network

Alike brain's neural network consisting of the smallest unit called "neurons", a deep learning technique named neural network also contains neurons ordered in layers to convert an incoming vector into outgoing results. This is done by neurons which applies a function to input and forwards the result to the next layer. On every passing some weights are applied to the signal which in turn the training phase to adopt a neural network to a problem. We simply provide all the extracted 40 features to the neural net consisting of 6 layers with the last layer as Sigmoid used for binary classification. Based on our observation Neural network has outperformed when epoch was 10 and batch size was 100 and uses Adam as an optimizer and binary cross entropy as loss function and provides optimum accuracy without overfitting.

3.6 Classification

After applying Machine Learning & Deep Learning techniques to train our model, we can classify whether the given email is a phished email or legitimate one. In output we can also see the accuracy with which the model is predicting whether the email is spam or ham. We can also choose a single model from various models of machine learning and deep learning using which we want to see the
Output and accuracy.

4. Result Analysis

We have tested our model with different datasets such as SpamAssassin and Ham/Spam email dataset.

4.1 Dataset Collected

An experiment is conducted in order to identify the input/output behaviour of the system. We have collected data from 2 different datasets. The datasets are SpamAssassin and spam/ham. These datasets are open-source and are freely available. The dataset collected in the experiment are
Identified and given in Table 4.1. Below table shows the total count of dataset and number of phished and legitimate emails present in those datasets which we have further used to train our model.

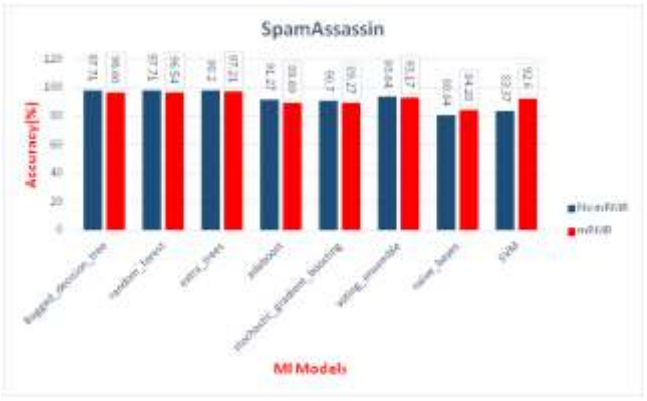
Table 4.1 Datasets and count of phished and legitimate emails

Dataset	Total Emails	Phished Emails	Legitimate Emails
Spam Assassin	6047	1897	4150
HSD	962	481	481

From the above table we can see that in the SpamAssassin dataset we have more legitimate emails than phished ones and HAM dataset have an equal number of phished and legitimate emails.

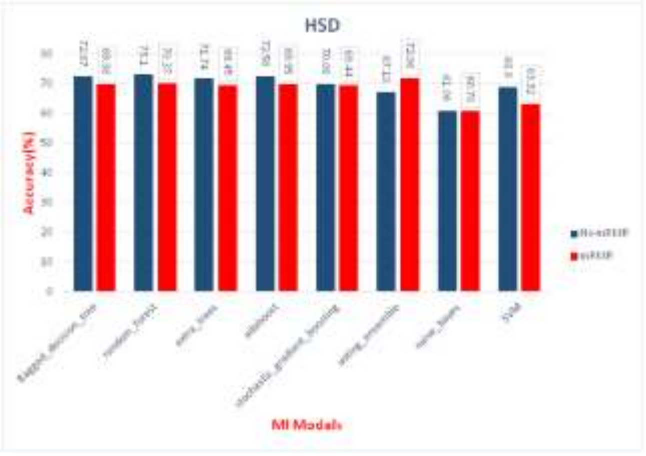
4.2 Result using Machine Learning Techniques.

Figure 4.2.1 Graph plot of Machine Learning models vs Accuracy for SpamAssassin dataset.



On testing various machine learning models with and without mRMR features (minimum redundancy maximum relevance) we observed a minimal difference in accuracy, we noticed that extra trees outperformed.

Figure 4.2.2 Graph plot of Machine Learning Models vs Accuracy for HSD dataset



On testing various machine learning models with and without mRMR features (minimum redundancy maximum relevance) we observed a minimal difference in accuracy, we noticed that random forest outperformed.

4.3 Result using Deep Learning Technique.

Figure 4.3.1 Epoch vs Accuracy plot for SpamAssassin dataset.

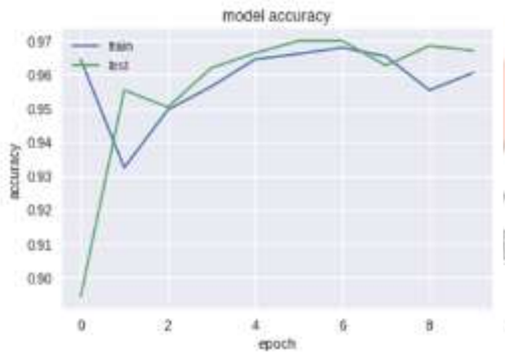
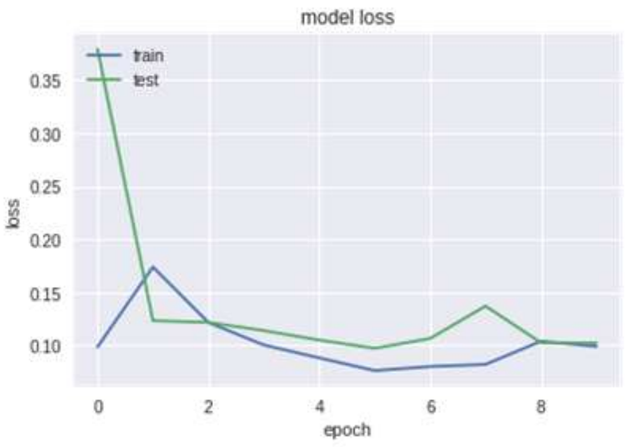


Figure 4.3.2 Epoch vs Loss plot for SpamAssassin dataset



4.4 Performance Evaluation

The standard of a domain system can be evaluated by comparing predictions obtained from a test set of known user ratings. These systems are typically measured using performance metrics such as precision, recall, F-measure and many others.

Precision: It is the measure of exactness, which determines the fraction of relevant items retrieved out of all items. Precision (P) is the proportion of legitimate emails that are truly legitimate.
P=TP/ (TP+FP) (1)
In our project,
TP = 5051, FP = 12
Precision= 5051/(5051+12)=0.99

Recall: It is a measure of completeness, which determines the fraction of relevant items retrieved out of all relevant items. It is the proportion of all legitimate emails.
R=TP/(TP+FN) (2)
In our project,
TP = 5051, FN= 0
Recall = 5051/ (5051+0) = 1.00

F-measure: It is defined as the HM (harmonic mean) of recall(R) and precision (P).
F-measure = (2*Precision*Recall) / (Precision Recall) (3)
In our project,
Precision = 0.99, Recall = 1.00
F-measure = (2*0.99*1.00)/(0.99+1.00) = 0.99

True Positive Rate: It is the percentage of phished mails from the total dataset that are correctly classified. Let P: the no of phished mails and Np: the no of correctly classified phished mails then true positive (TP) rate can be calculated as follows:
True Positive (TP) rate = Np / P (4)
In our project,
Np = 5051, P = 6846
TP rate = 5051/6846 = 0.73

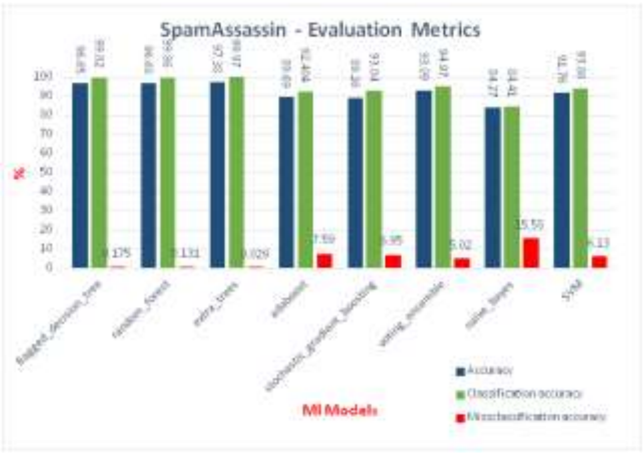
Confusion Matrix: This is the matrix which summarizes the predictions into 4 classes. True Positive, True Negative, False Positive and False Negative.

5051	12
0	1783

Here 5051 emails have been classified correctly as legitimate and 1783 out of 1795 emails are correctly classified as phished emails with 12 miss-classified emails as legitimate

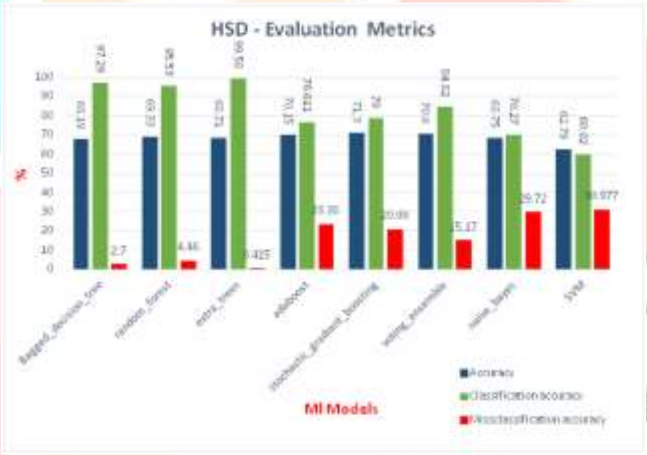
Classification accuracy: It is the no. of correct predictions divided by the total no. of predictions made, which when multiplied by 100 produces the Percentage (%).
Accuracy = (TP+TN) /P *100(%) (5)
In our project,
TP = 5051, TN = 1783, P=6846
Classification accuracy=(5051+1783)/6846*100 = 99.824%

Figure 4.4.1 Graph plot of evaluation metrics vs score for different ML models on SpamAssassin dataset.



The above figure depicts a graph plot of various machine learning models and the corresponding classification and misclassification accuracies acquired when implemented on SpamAssassin dataset. From the above graph we can observe that the models have high classification accuracy and low misclassification accuracy that depicts good model performance. Here we also observe that random forest model outperform as the classification accuracy is as high as 99.86 and misclassification accuracy is as low as 0.13.

Figure 4.4.2 Graph plot of evaluation metrics vs score for different ML models on HSD dataset.



The above figure depicts a graph plot of various machine learning models and the corresponding classification and misclassification accuracies acquired when implemented on
HSD dataset. From the above graph we can observe that the models have high classification accuracy and low misclassification accuracy. Here we also observe that extra-trees models outperform as the classification accuracy is as high as 99.58.

5. Conclusion

The objective of identifying phishing emails from legitimate one's is accomplished. In phishing detection, we take an email and then we categorize it into phished or legitimate emails so as to protect users from giving away the user sensitive information to the hackers. The Phishector takes input as an email and gives output as the accuracy with which it is predicting whether an email is phished or legitimate. We have worked upon 2 datasets: SpamAssassin and Ham-Spam datasets. Initially 40 features were extracted in order to apply Machine Learning models. Out of these 40 features, we have selected the important 5 and 8 features respectively. For Spamassassin dataset, we have observed that Extra trees outperforms amongst all 8 models with the accuracy of 97.21% and for Ham-Spam dataset, voting Ensemble gives the best accuracy of 72.06%. Also, we have used Deep Learning techniques (Neural Networks) with different combinations of epochs and batch size for predicting phishing emails. The SpamAssassin dataset yields the accuracy as 96.7%. The Ham-Spam dataset has the highest accuracy of 80%. Users can also choose which model he/she wants to apply to their email and can see the predictions of those models. We can also simultaneously compare outputs of multiple models and see which model is outperforming.

6. Acknowledgment

We are immensely obliged to Dr. Sharvari Govilkar for providing us with the understanding and idea of this project topic, for her indispensable support in garnering resources either by providing information or resources and also her supervision which made this project happen. We would like to offer our gratitude towards our college, Pillai College of Engineering, New Panvel and our principal Dr. Sandeep Joshi for providing us with all the facilities to comfortably and successfully complete our project. We would also take this opportunity to thank Dr. Sharvari Govilkar, H.O.D of Computer Department and other professors for their valuable support. Also, we would like to opine that it has indeed been a clinching experience for working out this project.. We would also like to express our grateful appreciation to all those people and staff who provided us the possibility to complete this report. So we take this chance to thank all those individuals who have helped us in visualizing and analysing this project.

References

[1] Adwan Yasin, Abdel Munem Abuhasan, “An Intelligent Classification Model for Phishing Email Detection”, Vol.8, No.4, *International Journal of Network Security & Its Applications (IJNSA)*, July 2016.

[2] Andronicus A. Akinyelu, Aderemi O. Adewumi, “Classification of Phishing Email using Random Forest Machine Learning”, Vol.2014, Article ID 425731, *Hindawi Publishing Corporation*, April 2014.

[3] Elif Yerli, Ibrahim Sogukpinar, “Email Phishing Detection and Prevention by using Data Mining Techniques”, *IEEE Xplore* November 2017.

[4] Fergus Toolan, Joe Carthy, “Phishing Detection using Classifier Ensembles,” *2009 eCrime Researchers Summit*, Tacoma, WA, USA, 2009.

[5] *Implementation*-https://github.com/TushaarGVS/Phishing.

[6] Minh Nguyen, Toan Nguyen, Thien Huu Nguyen, “A Deep Learning Model with Hierarchical LSTMs and Supervised Attention for Anti-Phishing”, *CEUR-WS* May 2018.

[7] Naghmeh Moradpoor, Benjamin Clavie, Bill Buchanan,”Employing Machine Learning Techniques for Detection and Classification of Phishing Emails,” *Computing Conference 2017*, London, UK, pp 149-156, July 2017.

[8] *Performance and Evaluation Metrics* https://www.ritchieng.com/machine-learning-evaluateclassification-model/

[9] Shivam Aggarwal, Vishal Kumar, S D Sudarsan, “Identification and Detection of Phishing Emails using Natural Language Processing Techniques”, *Proceedings of the 7th International Conference on Security of Information and Networks* September,2014.

[10] Simranjit Kaur Tuteja, Nagaraju Bogiri, “Email Spam Filtering using BPNN Classification Algorithm”, 2016 *International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT)*,2016.

[11] Sowndarya Karri, SSSN.Usha Devi N, “Framework for Phishing Detection in Email under Heave using Conceptual Similarity”, Vol.2 Issue:8, *International Journal on Recent and Innovation Trends in Computing and Communication (IJRITCC)*, August 2014.

[12] *SpamAssassindataset* - https://spamassassin.apache.org/

[13] *Spam/Hamdataset*-https://www.kaggle.com/balakishan77/spam-or-ham-email-classification/data

[14] Srishti Rawal, Bhuvan Rawal, Aakhila Shaheen, Shubham Malik, “Phishing Detection in Emails using Machine Learning,” Vol.12 – No. 7, *International Journal of Applied Information Systems (IJAIS)*, October 2017.

[15] Tianrui Peng, Ian G. Harris, Yuki Sawa, “Detecting Phishing Attacks using Natural Language Processing and Machine Learning,” *12th IEEE International Conference on Semantic Computing*, 2018